Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

- 1. (previously presented) A method for generating a chaos-based pseudorandom sequence (X_n) comprising the steps of:
 - defining a chaotic map for generating a pseudo-random sequence of integer numbers (x_n) comprised in a certain interval ([0, q]);
 - defining a function (H(x)) on a first interval $(x \in [0, q])$ whose inverse has a plurality of branches;
 - choosing a seed (x_0) of said pseudo-random sequence of integer numbers (x_n) comprised in said interval ([0, q]);
 - generating numbers of said pseudo-random sequence (x_n) ;
 - calculating numbers of a chaos-based pseudo-random sequence (X_n) by applying said function (H(x)) to corresponding integer numbers of said pseudo-random sequence (x_n); and
 - utilizing said chaos-based pseudo-random sequence (X_n) in an encryption application.
 - 2. (original) The method of claim 1, wherein the inverse of said function (H(x)) has a number of branches equal to the largest bound (q) of said interval ([0, q]).
 - 3. (original) The method of claim 1, wherein said chaotic map is a linear congruential generator.
 - 4. (original) The method of claim 3, wherein said linear congruential generator is defined by:
 - choosing a first integer number (m);

choosing a second odd integer number (p) greater than the power of 2 raised to said first integer number (2^m) ;

choosing a third integer number (M) much greater than said first integer number (m);

said chaotic map being defined by the following equation:

$$x_{n+1} = \left(\frac{p}{2^m} \cdot x_n\right) \mod 2^M.$$

- 5. (original) The method of claim 1, wherein defining said function (H(x)) comprises defining (H(x)) such that it may assume only two values $(\{0,1\})$.
 - 6. (original) The method of claim 5, comprising the steps of: representing in binary form said integer numbers (x_n) of said pseudo-random

sequence;

defining a second integer number k;

- defining said function (H(x)) as the binary sum of the k least significant bits of the binary representation of its argument (x).
- 7. (original) The method of claim 5, wherein said chaotic map is a truncated linear congruential generator.
- 8. (original) The method of claim 7, wherein said truncated linear congruential generator is defined by:

choosing a first integer number (m);

choosing a second odd integer number (p) greater than the power of 2 raised to said first integer number (2^{M});

choosing a third integer number (M) much greater than said first integer number (m);

said chaotic map being defined by the following equation:

$$x_{n+1} = trunc_k \left(\left(\frac{p}{2^m} \cdot x_n \right) \mod 2^M \right).$$

9. (original) The method of claim 7, wherein said linear congruential generator is defined by:

choosing a first integer number (m);

choosing a second odd integer number (p) greater than the power of 2 raised to said first integer number (2^m) ;

choosing a third integer number (M) much greater than said first integer number (m);

said chaotic map being defined by the following equations:

$$\begin{cases} y_n = x_n \oplus X_n \\ x_{n+1} = trunc_k \left(\left(\frac{p}{2^m} \cdot y_n \right) \mod 2^M \right) \end{cases}$$

- 10. (original) The method according to claim 4 wherein said third integer number (*M*) is greater than or equal to 64.
 - 11. (original) The method of claim 6, comprising the steps of: providing circuit means (MEM) for storing bit strings representing integer numbers (x_n) of said pseudo-random sequence;

providing a shift register (R1) coupled to said circuit means (MEM); storing a seed (x_0) in said circuit means (MEM); carrying out cyclically the following operations:

- copying in said shift register (R1) a bit string stored in the circuit means (MEM) representing a current number (x_n) of said pseudo-random sequence,
- providing k shift commands to said shift register (R1),
- generating a bit (X_n) of said chaos-based pseudo-random bit sequence by summing modulo 2 the k bits output by said shift register (R1),
- generating a bit string representing a successive number (x_{n+1}) of said pseudorandom sequence by summing up the bit string currently stored in said shift register (R1) and the bit string representing said current number (x_n) ,

- storing in the circuit means (MEM) the bit string representing said successive number (x_{n+1}) .
 - 12. (original) The method of claim 6, comprising the steps of:
 providing circuit means (MEM) for storing bit strings representing integer numbers (x_n) of said pseudo-random sequence;
 providing a register (R1) coupled to said circuit means (MEM);
 storing a seed (x₀) in said circuit means (MEM);
 carrying out cyclically the following operations:
- copying in said register (R1) a bit string stored in the circuit means (MEM) representing a current number (x_n) of said pseudo-random sequence,
- generating a bit (X_n) of said chaos-based pseudo-random bit sequence by summing modulo 2 the k least significant bits of the bit string stored in said register (R1),
- generating a bit string representing a successive number (x_{n+1}) of said pseudorandom sequence by summing up the bit string representing said current number (x_n) and the bit string obtained eliminating the k least significant bits of the bit string stored in said register (R1),
- storing in the circuit means (MEM) the bit string representing said successive number (x_{n+1}) .
- 13. (original) A generator of chaos-based pseudo random bit sequences, comprising:
 - circuit means (MEM) for storing bit strings representing integer numbers (x_n) of said pseudo-random sequence;
 - a register (R1) coupled to said circuit means (MEM);
 - an adder modulo 2 (XOR) summing the k least significant bits of the of the bit string stored in said register (R1), generating a bit (X_n) of said chaos-based pseudo-random bit sequence; and

- a second adder (ADD2) summing up the bit string representing said current number (x_n) and the bit string obtained eliminating the k least significant bits of the bit string stored in said register (R1).
- 14. (original) A generator of chaos-based pseudo random bit sequences, comprising:
- circuit means (MEM) for storing bit strings representing integer numbers (x_n) of said pseudo-random sequence;
- a shift register (R1) coupled to said circuit means (MEM);
- a command circuit (CONTROL) generating shift commands for said shift register (R1);
- second circuit means (R2) for storing the bits output by said shift register (R1); an adder modulo 2 (ADD1) summing the bits stored in said second circuit means (R2), generating a bit (X_n) of said chaos-based pseudo-random bit sequence; a second adder (ADD2) summing up the bit strings currently stored in said shift
- a second adder (ADD2) summing up the bit strings currently stored in said shift register (R1) and in said first circuit means (MEM), generating a bit string representing a successive number (x_{n+1}) of said pseudo-random sequence.
- 15. (previously presented) A method, comprising:
 generating a first pseudo-random value with a chaotic map;
 generating a first chaos-based pseudo-random value as a function of the first
 pseudo-random value, the function having an inverse with a plurality of branches; and
 utilizing the first chaos-based pseudo-random value in an encryption application.
- 16. (previously presented) The method of claim 15 wherein generating the first pseudo-random value comprises generating the first pseudo-random value within a finite interval of values.
 - 17. (previously presented) The method of claim 15 wherein:

generating the first pseudo-random value comprises generating the first pseudo-random value within a finite interval of values, the finite interval having an upper bound; and

the inverse of the function has a number of branches, the number being equal to the upper bound of the finite interval.

- 18. (previously presented) The method of claim 15, further comprising generating the first pseudo-random value from a seed value.
- 19. (previously presented) The method of claim 15, further comprising generating the first pseudo-random value from a previously generated pseudo-random value value.
- 20. (currently amended) The method of claim 15, further comprising generating the first pseudo-random value from a previous chaos-based pseudo-random value generated before the first chaos-based pseudo-random value.
- 21. (previously presented) The method of claim 15, further comprising: generating a second pseudo-random value from the first pseudo-random value with the chaotic map; and

generating a second chaos-based pseudo-random value as the function of the second pseudo-random value.

- 22. (previously presented) The method of claim 15 wherein the chaotic map comprises a linear congruential generator.
- 23. (previously presented) The method of claim 15 wherein the chaotic map comprises a truncated linear congruential generator.
- 24. (previously presented) The method of claim 15 wherein the function comprises an exclusive or function.

- 25. (previously presented) The method of claim 15 wherein the function comprises an exclusive or function of multiple bits of the first pseudo-random value.
 - 26. (previously presented) A circuit, comprising:
- a first generator operable to generate a first pseudo-random value with a chaotic map; and

a second generator coupled to the first generator and operable to generate a first chaos-based pseudo-random value as a function of the first pseudo-random value, the function having an inverse with a plurality of branches.

- 27. (previously presented) A system, comprising:
- a circuit, comprising,
- a first generator operable to generate a first pseudo-random value with a chaotic map; and

a second generator coupled to the first generator and operable to generate a first chaos-based pseudo-random value as a function of the first pseudo-random value, the function having an inverse with a plurality of branches.